SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Ivan Janković

Zagreb, 2014.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

UPRAVLJANJE ROBOTOM ANDROID APLIKACIJOM

Mentor: Student:

Prof. dr. sc. Mladen Crneković, dipl. ing. Ivan Janković

Zagreb, 2014.

Izjavljujem da sam ovaj rad izradio samostalno koristeći stečena znanja tijekom studija i navedenu literaturu.

Zahvaljujem se profesoru, mentoru dr.sc. Mladenu Crnekoviću što mi je omogućio pristup robotu kao i pomoć prilikom izrade ovog rada.

Ivan Janković

SVEUČILIŠTE U ZAGREBU

FAKULTET STROJARSTVA I BRODOGRADNJE

Središnje povjerenstvo za završne i diplomske ispite

Povjerenstvo za završne ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum	Prilog
Klasa:	
Ur.broj:	

ZAVRŠNI ZADATAK

Student: **IVAN JANKOVIĆ** Mat. br.:0035185242

Naslov rada na hrvatskom jeziku: **UPRAVLJANJE ROBOTOM ANDROID APLIKACIJOM**

Naslov rada na engleskom jeziku: **ANDROID APPLICATION CONTROLLED ROBOT**

Opis zadatka:

Zbog svoje raširenosti i pristupačnosti uređaji s Android operativnim sustavom (mobilni telefoni, tableti) postaju sve popularniji za upravljanje i nadzor procesa. Povezivanje s procesom ostvaruje se bežično preko Wi-Fi ili bluetoothom vezom.

Potrebno je projektirati grafičko sučelje koje će omogućiti pokretanje robota RM501 Android aplikacijom u realnom vremenu.

Tražena rješenja:

- opisati naredbe odabranog robota kojima će korisnik pokretati robota,
- dizajnirati grafičko sučelje za pokretanje robota,
- definirati vezu između Android aplikacije i kinematičkog kontrolera robota.

Zadatak zadan:
11. studenog 2013.

Zadatak zadao:

Prof. dr. sc. Mladen Crneković

Rok predaje rada:
1. rok: 21. veljače 2014.
2. rok: 12. rujna 2014.

Predviđeni datumi obrane:
1. rok: 3., 4 . i 5. ožujka 2014.
2. rok: 22., 23. i 24. rujna 2014.

Predsjednik Povjerenstva:

Prof. dr. sc. Zoran Kunica

SADRŽAJ

POPIS SLIKA

POPIS TABLICA

POPIS OZNAKA

Oznaka	Opis oznake
p_x	Pozicija prihvatnice
p_y	Pozicija prihvatnice
p_z	Pozicija prihvatnice
φ	Orijentacija prihvatnice (nagib)
ψ	Orijentacija prihvatnice (valjanje)
q_1	Upravljane koordinate robota (struk)
q_2	Upravljane koordinate robota (rame)
q_3	Upravljane koordinate robota (lakat)
q_4	Upravljane koordinate robota (nagib)
q_5	Upravljane koordinate robota (valjanje)

SAŽETAK

U današnje se vrijeme u industriji i ostalim pogonima zbog sve veće potrebe koriste roboti. Osim u industriji, roboti se koriste i u obrazovne svrhe. Robot se većinom programira i upravlja preko računala, ali postoje i drugi načini za upravljanje robotom. Zbog svoje raširenosti i pristupačnosti, uređaji s Android operativnim sustavom (mobilni telefoni, tableti) postaju sve popularniji za upravljanje i nadzor procesa. U ovom je radu opisana izrada i projektiranje grafičkog sučelja aplikacije koja omogućuje upravljanje robotom Mitsubishi RM501. Robotom se upravlja mobilnim uređajem preko Bluetooth veze koja omogućuje komunikaciju preko kinematičkog kontrolera pomoću određenih naredbi.

Ključne riječi: Android; Bluetooth; Robot; Android aplikacija; Mobilni uređaj

SUMMARY

Due to the growing needs, industry nowadays uses robots. Except in the industry, robots are also used for educational purposes. Robot is mostly programmed and controlled via computer, but there are other ways to control it. Devices with Android operating system (mobile phones, tablet) are becoming increasingly popular for the management and oversight of the process because of theirs prevalence and accessibility. In this paper, I will describe drafting and designing GUI application that allows to operate the robot Mitsubishi RM501. The robot is controlled by mobile device via Bluetooth connection that allows communication via kinematic controller using certain commands.

Key words: Android; Bluetooth; Robot; Android application; Mobile phone

1. UVOD

U današnje vrijeme zbog sve veće potrebe u industriji i ostalim pogonima koriste se roboti. Roboti u industriji koriste se za mnogo različitih potreba kao što su zavarivanje, posluživanje strojeva, montaža nekih dijelova te na kraju za završna mjerenja i kontrolu da bi svaki proizvod bio jednak i izrađen u granicama tolerancija. Roboti danas uvelike zamjenjuju ljude u mnogim poslovima zbog povećanja kvalitete proizvoda, radnih operacija koje su izvan fizičkih mogućnosti čovjeka te na kraju iz ekonomskih razloga(povećanje produktivnosti, povećanje proizvodnje i uštede energije). Zbog njihove današnje raširenosti i načina upravljanja potrebni su i ljudi koji će raditi s robotima. Zbog toga se danas roboti koriste i u obrazovne svrhe, kako bi ljudi naučili na koji se način upravlja i programira robot. Robot se većinom programira i upravlja preko računala ali u današnje vrijeme postoje i drugi načini za upravljanje robotom. Zbog svoje raširenosti i pristupačnosti mobilni telefoni i tableti s Android operativnim sustavima su jedan od načina na koji se može upravljati robotom i nadzirati proces. Pomoću Android aplikacije moguće je na jednostavan način upravljati robotom. U ovom završnom radu bit će riječi o robotu kojim je upravljano uz pomoć Android aplikacije, također će biti riječi o Android platformi za mobilne telefone i tablete te o Android aplikaciji. Grafičko sučelje Android aplikacije koje omogućava pokretanje robota rađeno je u programskom paketu Android SDK. Veza između mobilnog telefona i robota ostvaruje se bluetooth vezom.

2. ROBOT MITSUBISHI RM501

Prije same izrade Android aplikacije sa grafičkim sučeljem koje omogućuje pokretanje navedenog robota bilo je potrebno upoznavanje sa navedenim robotom. Kao i svaki proizvod tako i svaki robot ima neke svoje određene karakteristike po kojima se razlikuje od drugih robota. Neke od karakteristika koje u nekim slučajevima imaju veliku važnost su nosivost robota, broj stupnjeva slobode gibanja, struktura robota, način upravljanja i programiranja te tako i njihova cijena. Upoznavanje s nekim od navedenih karakteristika robota je bilo važno pri izradi Android aplikacije jer se na temelju tih karakteristika morala prilagoditi i sama izrada aplikacije. Robot Mitsubishi RM501 ima pet stupnjeva slobode gibanja i mogućnost hvatanja i držanja predmeta s dva prsta. Robot je manjih dimenzija te je takav dobar za korištenje u obrazovne svrhe.

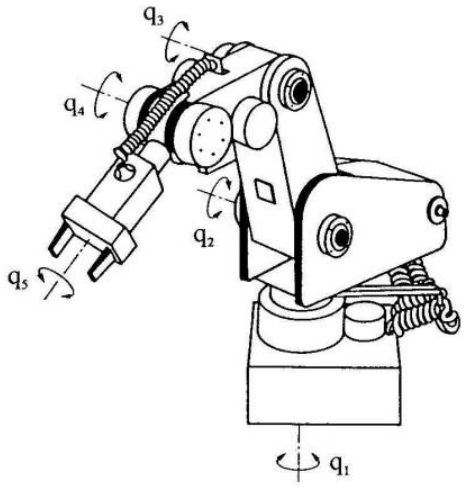
Slika 1. Robot Mitsubishi RM501

2.1 Karakteristike robota

Kao što je već navedeno prilikom izrade Android aplikacije važno je znati neke od karakteristike robota. Položaj prihvatnice može se opisati pomoću vektora vanjskih i unutarnjih koordinata. Vektor vanjskih koordinata je $\boldsymbol{r} = [p_x\ p_y\ p_z\ \varphi\ \psi\ ]^T$ a vektor unutarnjih koordinata je $\boldsymbol{q} = [q_1\ q_2\ q_3\ q_4\ q_5\ ]^T$. Pomoću kinematike robota moguće je dobiti vezu između unutarnjih i vanjskih koordinata odnosno vektora **q** i **r.**

2.1.1 Robotska ruka:

- Pet stupnjeva slobode gibanja:
  - q_1 = struk 300°
  - q_2 = rame 130°
  - q_3 = lakat 90°
  - q_4 = nagib $\pm$90°
  - q_5 = valjanje $\pm$180°
- Nosivost robota: 1.2 kg
- Pogon robota: istosmjerni servomotor (24 V, 15 W)
- Indirektno mjerenje položaja robota
- Maksimalna brzina prihvatnice 400 mm/s
- Prijenos pogona robota: zupčanici, remen
- Programiranje gibanja
- Težina cijelog robota: 27 kg [2]



Slika 2. Broj stupnjeva slobode gibanja

2.1.2 Upravljačka jedinica

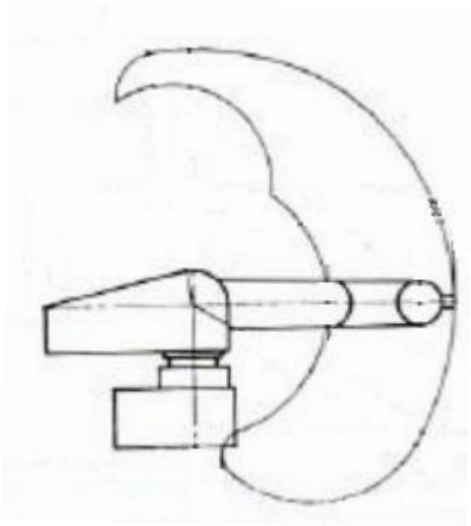
- Energetski i informatički dio
- Veza s robotom preko dva kabela (energija i informacije)
- Priključci:
    - Serijski RS232C
    - Privjesak za učenje
    - Paralelni printerski ulaz
- Procesor Z80
- Test način rada [2]

2.1.3 Prihvatnica

- Masa: 0.7 kg
- Maksimalna otvorenost 60 mm
- Maksimalna sila 44 N [2]

2.1.4 Radni prostor robota

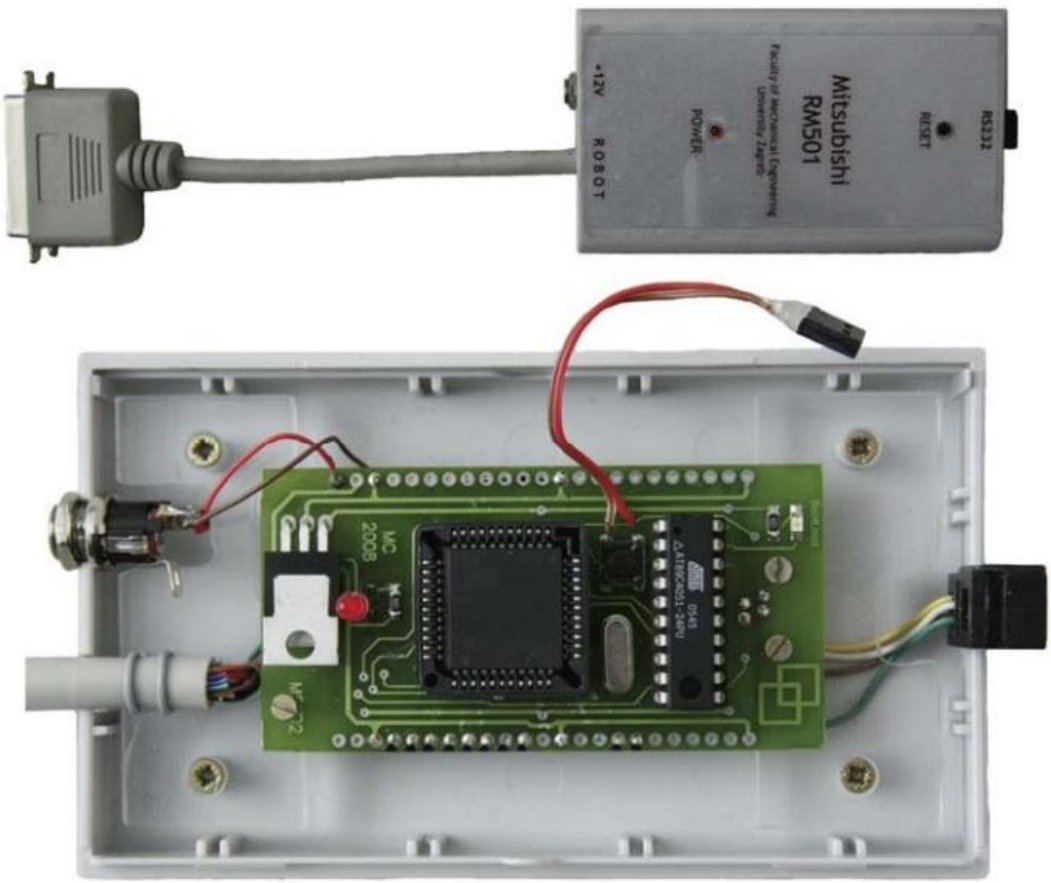
Radni prostor robota je prostor u kojem se giba prihvatnica. [3]



Slika 3. Radni prostor robota

2.2 Kinematički kontroler

Robot Mitsubishi RM501 radi preko paralelne komunikacije što je danas način koji se jako rijetko koristi. Spajanjem računala ili bilo kakvog novijeg uređaja nije bila moguća komunikacija s robotom. Pomoću kinematičkog kontrolera moguće je spojiti bilo kakvo računalo ili noviji uređaj koji radi preko serijske komunikacije. Kinematički kontroler radi tako da serijsku komunikaciju koja dolazi od strane računala pretvara u paralelnu komunikaciju preko koje radi robot. Kinematičkim kontrolerom možemo kontrolirati pokrete robota uz nove naredbe. Kontroler je izrađen pomoću Atmelovog mikrokontrolera AT89C5ID2. [1]



Slika 4. Kinematički kontroler

2.3 Naredbe za upravljanje robotom

Korištenjem kinematičkog kontrolera moguće je koristiti više naredbi koje su vezane za upravljanje robotom Mitsubishi RM501 pa tako i izravno postavljanje vanjskih koordinata robota. Prilikom korištenja nekih od naredbi kinematički kontroler odgovori na naredbu ili signalizira da je došlo do pogreške. Neke od naredbi dane su u [Tablica 1.].

Tablica 1. Naredbe za komunikaciju

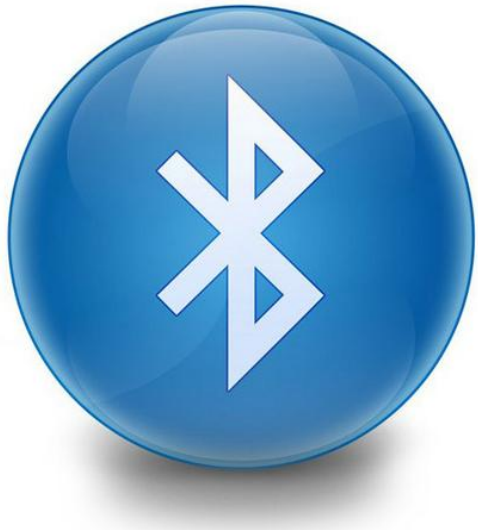
Broj naredbe	Naredba	Odgovor	Opis
1	x127y-125z72.3f33p20.1	#Ex	Apsolutna pozicija robota
2	Dx10.5y20z-10f0p33	#Ex	Relativna pozicija robota
3	OPEN	#E0	Otvaranje hvataljke
4	CLOSE	#E0	Zatvaranje hvataljke
5	INIT	#E0	Inicijalizacija robota
6	HOME	#Ex	Home pozicija
7	?INIT	#INIT=NO ili YES	Status inicijalizacije
8	?R	#R= x127y-125z72.3f33p20.1	Vektor **r**

Tablica 2. Značenje pogrešaka #Ex

Značenje pogrešaka #Ex	
x	Opis
0	Nema pogreške
1	Parametar a1 izvan dosega
2	Parametar a2 izvan dosega
3	Parametar a3 izvan dosega
4	Parametar a4,a5 izvan dosega
5	Ulazni parametar izvan dosega
6	Točka izvan dosega robota
7	Robot nije inicijaliziran
8	Opasnost od sudara
9	Pogreška u naredbi

3. BLUETOOTH VEZA

Android aplikacija koja je instalirana na mobilnom uređaju preko bluetooth veze upravlja robotom Mitsubishi RM501. Bluetooth je naziv za bežični prijenos podataka i govora, namijenjen za malu potrošnju i jeftine bežične komunikacije na manje udaljenosti.[4] Bluetooth tehnologija razvijena je za bežično povezivanje (RF veze kratkog dosega) prijenosnih i stolnih računala, mobilnih telefona, kamera i drugih digitalnih uređaja na kratkim udaljenostima. Udaljenosti na kojima se komuniciraju uređaji su do 10 m (snaga odašiljanja 1mW), a s većom snagom odašiljača mogu biti i do 100 m. [9]



Slika 5. Simbol Bluetootha

U svojoj osnovi Bluetooth se ostvaruje mikročipom koji koristi radio prijenos kratkog dosega (nije potrebna optički vidljiva linija) za prijenos informacija. Mikročip se ugrađuje u uređaj (kamera, tipkovnica, mobilni telefoni i ostali digitalni uređaji) ili se spaja preko univerzalne serijske sabirnice (eng. USB – Universal Serial Bus) serijskog priključka ili PC kartice. Brzine prijenosa podataka su do 1Mb/s. Koristi se ISM (Industrial – Scientific - Medicine) nelicenciran frekvencijski pojas od 2.4 GHz do 2.48 GHz i koristi FHSS modulacijsku tehniku. Za korištenje ISM pojasa nije potrebna dozvola i ne plaća se naknada. Bluetooth specifikacija definira Bluetooth protokol stack za prijenos podataka.

Bluetooth komunikacijom se izgrađuju takozvane piconet mreže koje se sastoje od dva ili više bežično povezanih (ad-hoc) uređaja. Unutar svakog piconet-a postoje više „slave“ jedinica i jedna „master“ jedinica koja upravlja njima.

Slika 6. Bluetooth modul

Kada se Bluetooth uređaji nađu unutar dometa oni uspostavljaju ad hoc mrežu. U toj mreži jedan od uređaja postaje nadređen (master), a svi ostali uređaji su podređeni (slave). Važno je napomenuti da bilo koji uređaj može postati nadređenim. Uređaj koji uspostavlja vezu, prema definiciji, preuzima vezu.

3.1 Zaštita i sigurnost Bluetooth tehnologije

Bluetooth tehnologija implementira sigurnosne protokole. Ti protokoli su definirani na nižim razinama Bluetooth stack protokola. Razine zaštite su:

1. Svaki Bluetooth uređaj ima jedinstvenu IEEE MAC adresu (48 bita),
2. Primjena frekvencijskog preskakivanja (FHSS) i mala snaga emitiranja,
3. Bluetooth sigurnosni protokol zahtijeva 2 tajna ključa. [9]

4. ANDROID PLATFORMA

Android je prvi otvoreni operacijski sustav za mobilne uređaje (mobilni telefoni, tableti, netbook računala) pokrenut od strane Google Inc. i vođen od strane OHA (Open Handset Alliance) grupe koja danas broji preko 80 tehnoloških kompanija između kojih se nalaze HTC, Samsung, Motorola. Android je zasnovan na jezgri Linux 2.6 i napisan u C++ programskom jeziku.[10] Android je potpuno prilagodljiva platforma za mobilne uređaje. Sadrži operacijski sustav, opremu (softver koji djeluje između aplikacije i mreže) i ključne mobilne aplikacije. Sadrži veliki broj API-a (application programming interface) sučelje za programiranje aplikacija koji omogućuje neovisnim proizvođačima izradu njihovih aplikacija. Android je izgrađen tako da omogućava programerima stvaranje aplikacija koje u potpunosti koriste sve što uređaj nudi. Većina aplikacija pisana je u java programskom jeziku koristeći SDK (Android Software Development Kit).[10] Aplikacije je moguće pisati i u C++ programskom jeziku. Ovakvim postupkom aplikacije se ubrzavaju i do nekoliko puta, no pisanje samog programa je puno složenije. Takvom izradom aplikacija, Android pomiče granice stvaranja novih i inovativnih aplikacija, kao na primjer programer može kombinirati informacije s weba sa podacima na uređaju pojedinog korisnika, kao što su kalendar, vrijeme ili zemljopisna lokacija.

Slika 7. Logo Android platforme

Android platforma dolazi s mnoštvo već izrađenih aplikacija gdje se za izradu aplikacija koristio Java programski jezik. Neke od aplikacija su:

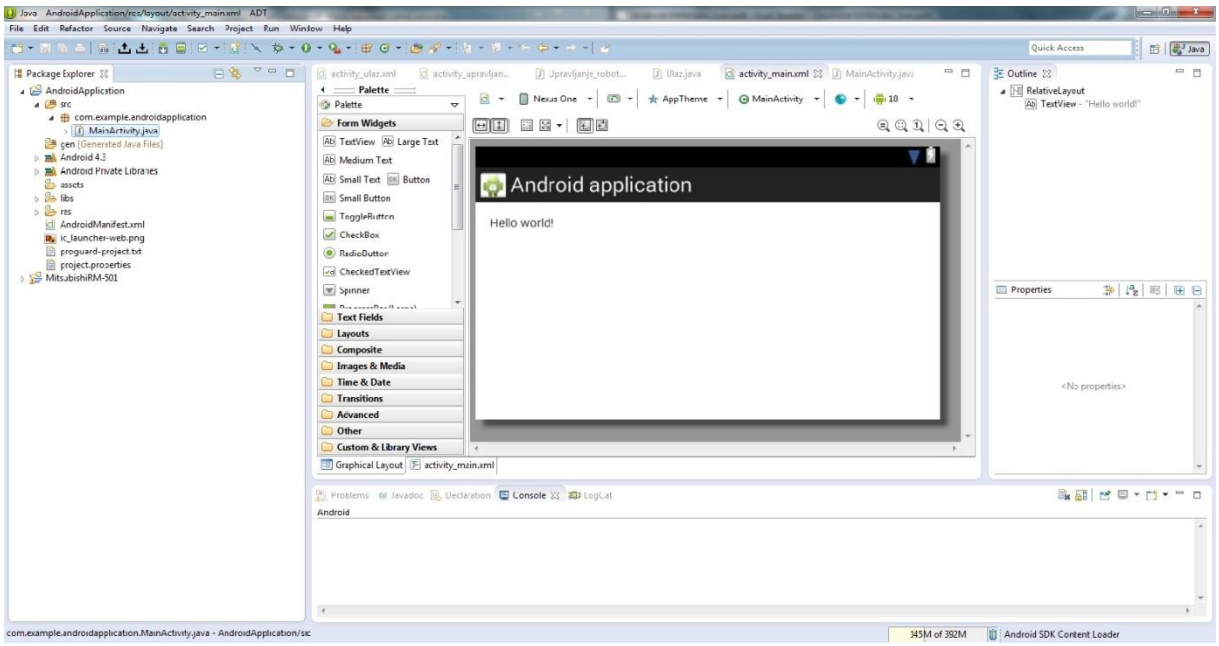
- Home – prikazuje aplikacije koje su već instalirane na mobilnom uređaju, te prikazuje grafičke elemente
- Email – osigurava pristup poslužiteljima e-maila
- Web Browser – pretraživač za Internet
- Calculator, Calendar, Camera i mnoge druge aplikacije

U današnje vrijeme android platforma se razvija iz dana u dan sa novim aplikacijama i novim tehnologijama koje omogućuju korištenje mobilnih telefona već kao i računala. Jedna od njegovih glavnih prednosti je dobra organizacija, koja ima potencijal da iskoristi svu moć i znanje zajednice otvorenog koda. Danas najnovija verzija Android platforme je 4.4.2 KitKat koja omogućuje puno brži rad mobilnih uređaja i korištenje svih vrsta aplikacija. [10]

Slika 8. Android 4.4.2 KitKat

5. IZRADA ANDROID APLIKACIJE

Android omogućuje svim neovisnim korisnicima izradu svojih Android aplikacija. Većina aplikacija pisana je u java programskom jeziku koristeći SDK (Android Software Development Kit). Preporučeni način za razvijanje Android aplikacije je korištenje Eclipsea s dodatkom ADT (Android Development Tools) koji je sadržan u SDK. ADT omogućuje uređivanje, razvoj i otklanjanje grešaka integrirane izravno u IDE (Integrated development environment). Aplikacije za Android pisane su u programskom jeziku Java, prevedeni Java kod zajedno sa svim podacima i datotekama resursa potrebnim za aplikaciju je zapakiran u Androidov paket, tj. datoteku koja ima format .apk. Prvi korak u programiranju za Android aplikacije jest preuzimanje SDK (Android Software Development Kit). SDK uključuje sveobuhvatan skup razvojnih alata. Nakon preuzimanja i instalacije SDK-a pokrećemo Eclipse u kojem izrađivamo aplikaciju.



Slika 9. Eclipse

Eclipse ima izbornik u kojem se nalaze različiti dijelovi koji nam koriste za upravljanje aplikacijom. Dijelovi kao što su tipke, različiti tekstovi, umetanje vremena i datuma, šifre kod pokretanja aplikacije, slika, kalendara te umetanje različitih alata za povećavanje ili smanjivanje slike. Svi dijelovi iz izbornika u radni prostor stavljaju se na način „drag and drop“. Kad postavimo sve potrebne dijelove u radni prostor tada je potrebno

sve dijelove povezati tj. napisati program koji će kad pritisnemo tipku nešto izvršiti. Taj program se piše u programskom jeziku Java. Nakon što smo napisali program potrebno ga je isprobati pomoću Android Emulatora. Android Emulator je virtualni prikaz mobilnog uređaja gdje vidimo dali ima kakvih grešaka u aplikaciji prije nego što izbacimo aplikaciju i instaliramo je na naš mobilni uređaj. Pošto smo mi sami izrađivali aplikaciju na mobilnom uređaju je potrebno podesiti neke postavke kako bi smo uspješno instalirali našu aplikaciju. Kod sigurnosti na našem mobilnom uređaju potrebno je dopustiti instalaciju aplikacija iz drugih izvora.

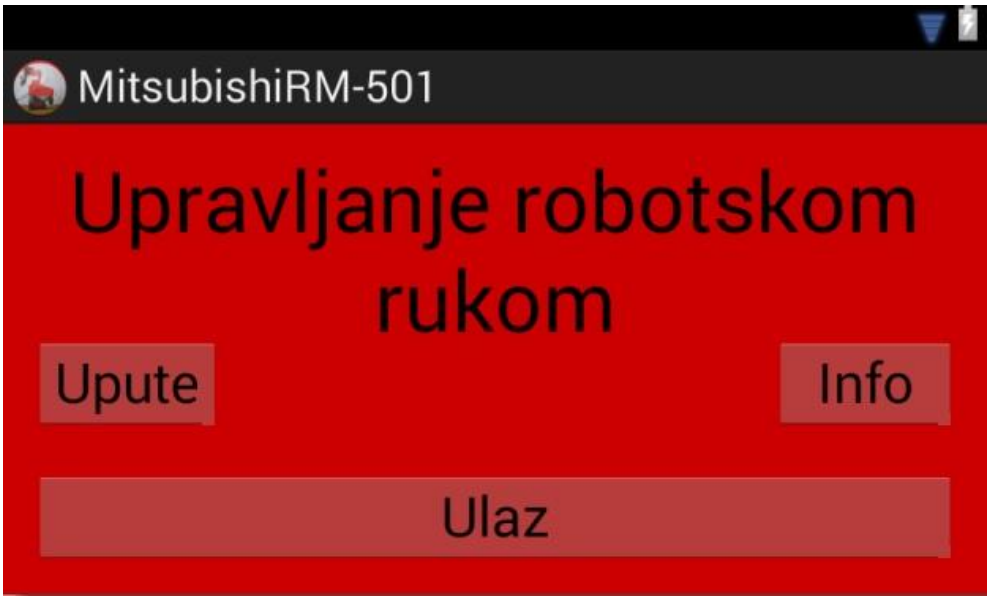
5.1 Android aplikacija za upravljanje robotom

Prije početka izrade aplikacije potrebno je definirati ime aplikacije, ikonu po kojoj će se prepoznati aplikacija te Android verziju kojom će se moći pokrenuti aplikacija. Ova aplikacija radi na verziji Android 4.1.2 te svim novijim Android verzijama. Nakon odabira početnih parametara počinjemo oblikovati našu aplikaciju.

Slika 10. Ikona aplikacije

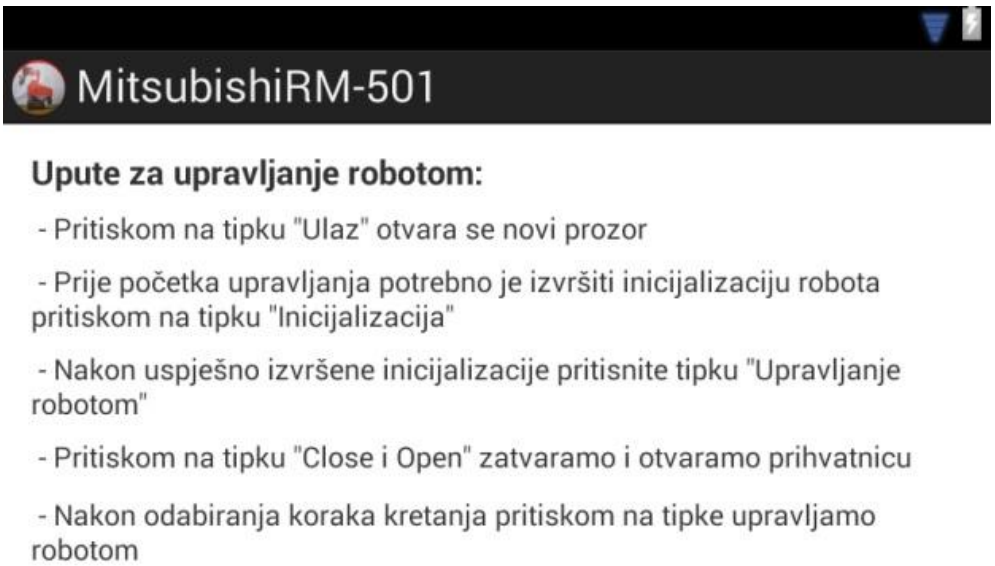
Aplikacija se sastoji od više prozora u kojima su smještene tipke kojim otvaramo nove prozore te tipke za upravljanje robotom. U glavnom prozoru pritiskom na tipke „Upute“ možemo pročitati upute gdje je navedeno na koji način se upravlja robotom. Pritiskom na tipku „Info“ otvara nam se prozor gdje su navedene karakteristike robota. Nakon pročitanih uputa i karakteristika robota potrebno je izvršiti spajanje sa robotom preko Bluetootha. Pritiskom na tipku „Menu“ otvara nam se izbornik gdje pritiskom na tipku „Spoji“ pokrećemo listu za traženje svih uređaja koji imaju mogućnost spajanja preko Bluetootha. Kad pronađemo naš Bluetooth uređaj pritiskom na njega uspostavljamo vezu. Nakon što smo se

uspješno spojili preko Bluetooth veze u glavnom prozoru je potrebno pritisnuti tipku „Ulaz“ gdje se pritiskom na tipku otvara novi prozor .



Slika 11. Početni prozor aplikacije

Pritiskom na tipku „Upute“ otvara se prozor gdje možemo pročitati upute koje su nam potrebne kako bi upravljali robotom

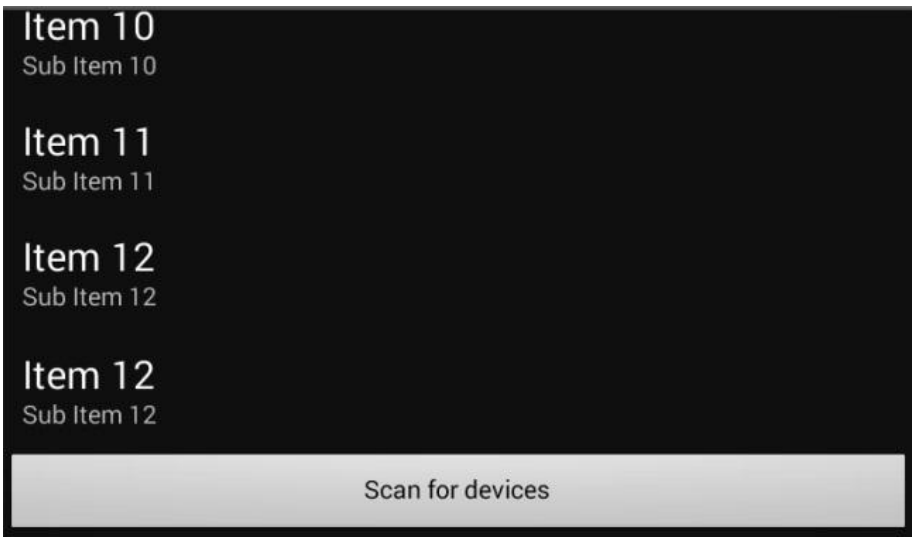


Slika 12. Upute za upravljanje robotom

Pritiskom na tipku „Info“ otvara se prozor gdje možemo pročitati karakteristike robota kojim upravljamo.

Slika 13. Karakteristike robota

Spajanje aplikacije sa robotom uspostavlja se preko Bluetooth veze pritiskom tipke „Menu“. Otvara se mali izbornik gdje možemo odabrati dvije opcije. Jedna od opcija je spajanje tj. traženje uređaja kojim je uključen Bluetooth. Druga opcija je postavljanje vidljivosti na 300 sekundi kako bi naš uređaj bio vidljiv drugim uređajima i kako bi se drugi uređaj mogao spojiti s našim uređajem. Kod izbora prve opcije pritiskom na tipku „Spoji“ otvara se prozor gdje možemo odabrati uređaj s kojim želimo uspostaviti Bluetooth vezu.

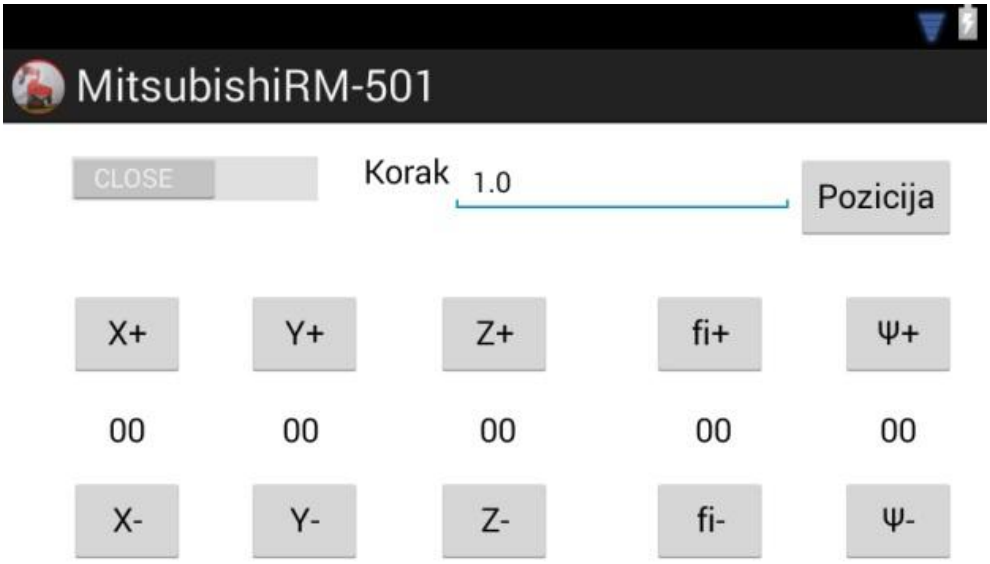


Slika 14. Lista vidljivih uređaja

Nakon što smo odabrali odgovarajući uređaj provjerimo Bluetooth status koji se nalazi u lijevom kutu prozora, da smo sigurni da se naš uređaj tj. aplikacija uspostavila vezu s robotom. Kad Bluetooth status potvrdi uspješno spajanje na Bluetooth modulu crvena lampica će se promijeniti u žutu boju. Nakon toga pritisnemo tipku „Ulaz“ za otvaranje novog prozora. U novom prozoru smještene su dvije tipke. Prije nego što počnemo upravljati robotom, robot moramo inicijalizirati. Pritiskom na tipku „Inicijalizacija“ šaljemo robotu naredbu za inicijalizaciju. Inicijalizacija je proces radnji koje se obavljaju prije početka izvršavanja programa ili upravljanja, sa ciljem da se utvrdi ispravnost svih potrebnih konstanti, ili indikacija, kako bi se utvrdila ispravnost uređaja a time i kako bi se mogao uspješno izvršiti program ili upravljanje. Aplikacija je izrađena na način da stalno komunicira s robotom te tako prilikom prvog pokretanja robota pritiskom na tipku „inicijalizacija“ robot će izvršiti inicijalizaciju. Ponovnim pritiskom na tipku robot neće izvršiti inicijalizaciju zbog toga jer je inicijalizacija već izvršena te će nam vratiti obavijest „Robot je već inicijaliziran“. Kod svakog novog pokretanja robota potrebno je izvršiti inicijalizaciju.

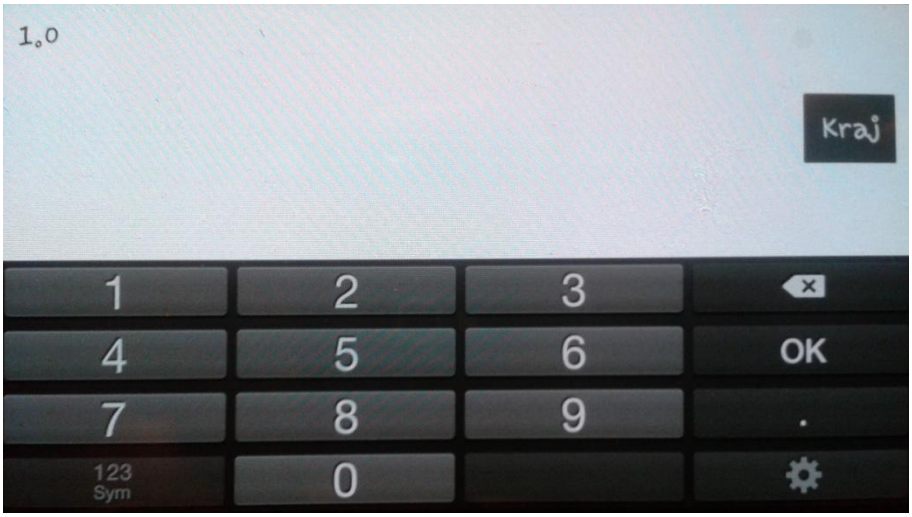
Slika 15. Inicijalizacija robota

Nakon izvršene inicijalizacije robota možemo upravljati robotom. Pritiskom tipke „Upravljanje robotom“ otvara nam se novi prozor. U novom prozoru se nalaze tipke kojima upravljamo pokretima robota. Tipka „Open i Close“ otvaraju i zatvaraju hvataljku. Moguće je i podešavanje koraka gibanja robota pritiskom na to predviđeno mjesto. Pritiskom na tipku „Pozicija“ aplikacija nam vraća trenutni položaj robota te ispisuje položaj za svaku od koordinata. Ako postavimo preveliki korak gibanja ili ako dođemo do graničnog položaja javi nam se pogreška te nam se na ekran ispisuje svaka od grešaka. Pogreške koje se javljaju navedene su u [tablici2].



Slika 16. Prozor za upravljanje robotom

Pritiskom na za to predviđenom mjesto otvara nam se prozor u koji upisujemo željeni korak gibanja.



Slika 17. Korak gibanja

Kod izrađivanja aplikacije u Eclipse-u, izgled prozora uređivamo po želji. Postavljanjem tipki i naziva tipki te ostalih tekstova koji nam služe kao smjernice za lakše snalaženje u aplikaciji program sam stvara dio programskog koda. Pokretanjem Eclipsea tj. određivanjem početnih parametara program stvori početni dio programskog koda koji mi kasnije proširujemo. Na prikazanoj slici nalazi se prozor koji je uređen po želji. [slika 18.]

Slika 18. Prozor uređivan po želji

Kad smo uredili prozor tipkama, različitim tekstovima, bojama pozadine program je sam stvorio dio programskog koda. Prilikom uređivanja prozora stvaranje programskog koda se ne vidi već program to radi u pozadini. Dolje je prikazan dio programskog koda koji je program sam stvorio za navedeni prozor.

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@android:color/holo_red_dark"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >
```

```
<Button
    android:id="@+id/Info"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignRight="@+id/textView1"
    android:layout_alignTop="@+id/Upute"
    android:minWidth="100dp"
    android:text="Info"
    android:textSize="30sp"
    android:typeface="normal" />

<Button
    android:id="@+id/Upute"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/Ulaz"
    android:layout_alignLeft="@+id/textView1"
    android:layout_marginBottom="21dp"
    android:minHeight="50dp"
    android:minWidth="80dp"
    android:text="Upute"
    android:textColor="@android:color/background_dark"
    android:textColorHint="@android:color/background_dark"
    android:textSize="30sp"
    android:typeface="normal" />

<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true"
    android:gravity="center_horizontal"
    android:text="@string/Naslov"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:textSize="45sp" />

<Button
    android:id="@+id/Ulaz"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/Upute"
    android:layout_alignParentBottom="true"
    android:layout_alignRight="@+id/Info"
    android:text="Ulaz"
    android:textSize="30sp"
    android:typeface="normal" />

 </RelativeLayout>
```

Drugi dio programskog koda koji piše korisnik je malo drugačiji. Programski jezik kojim pišemo ostali dio aplikacije je Java. Tim dijelom programskog koda povezujemo tipke i ostale prozore koje želimo otvoriti pritiskom na neku od tipki. Taj dio koda sastoji se od niza naredbi koji se koriste u programskom jeziku Java. Odabirom početnih parametara program već sam stvara dio programskog koda koji je napisan u programskom jeziku Java. Dolje se nalazi primjer programskog koda koji je stvoren na samom početku.

```
package com.example.robot;


import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;


public class MainActivity extends Activity {


	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_main);
	}


	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
		// Inflate the menu; this adds items to the action
		bar if it is present
		getMenuInflater().inflate(R.menu.main, menu);
		return true;


	}

}
```

Dio programskog koda koji je namijenjen za povezivanje aplikacije sa robotom preko Bluetooth veze.

```
package com.example.mitsubishirm_501;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.UUID;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Context;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;

public class BluetoothService  {
      // Debugging
      private static final String TAG = "BluetoothChatService";
      private static final boolean D = true;
      // Unique UUID for this application
      private static final UUID MY_UUID = UUID
                  .fromString("00001101-0000-1000-8000-00805F9B34FB");

      // Member fields
      private final BluetoothAdapter mAdapter;
      private  Handler mHandler;

      private ConnectThread mConnectThread;
      private ConnectedThread mConnectedThread;
      private int mState;

      // Constants that indicate the current connection state
      public static final int STATE_NONE = 0; // we're doing nothing
      public static final int STATE_LISTEN = 1; // now listening for
incoming
                                                // connections
      public static final int STATE_CONNECTING = 2; // now initiating
an outgoing
                                                // connection
      public static final int STATE_CONNECTED = 3; // now connected
to a remote
                                                // device

      public BluetoothService(Context context, Handler handler) {
            mAdapter = BluetoothAdapter.getDefaultAdapter();
            mState = STATE_NONE;
```

```
            mHandler = handler;

      }

      public void setBluetoothServiceHandler(Handler handler){
            mHandler = handler;
      }



private synchronized void setState(int state) {
            //Postavljanje statusa
            if (D)
                  Log.d(TAG, "setState() " + mState + " -> " + state);
            mState = state;

            // Give the new state to the Handler so the UI Activity
can update
            mHandler.obtainMessage(MainActivity.MESSAGE_STATE_CHANGE,
state, -1)
                        .sendToTarget();
      }


      public synchronized int getState() {
            return mState;
      }


      public synchronized void start() {
            if (D)
                  Log.d(TAG, "start");
// Cancel any thread attempting to make a connection
            if (mConnectThread != null) {
                  mConnectThread.cancel();
                  mConnectThread = null;
            }

            // Cancel any thread currently running a connection
            if (mConnectedThread != null) {
                  mConnectedThread.cancel();
                  mConnectedThread = null;
            }

            // Start the thread to listen on a BluetoothServerSocket

            setState(STATE_LISTEN);
      }
```

```
	public synchronized void connect(BluetoothDevice device) {
		//Stvaranje objekta klase ConnectThread i povezivanje sa
uređajem
		if (D)
			Log.d(TAG, "connect to: " + device);


// Cancel any thread attempting to make a connection
		if (mState == STATE_CONNECTING) {
			if (mConnectThread != null) {
				mConnectThread.cancel();
				mConnectThread = null;
			}
		}

		// Cancel any thread currently running a connection
		if (mConnectedThread != null) {
			mConnectedThread.cancel();
			mConnectedThread = null;
		}

		// Start the thread to connect with the given device

		mConnectThread = new ConnectThread(device);

		mConnectThread.start();
		setState(STATE_CONNECTING);
	}


	public synchronized void connected(BluetoothSocket socket,
			BluetoothDevice device) {
	//Kada je povezivanje uspostavljeno napravi objekt klase
	ConnectedThread
		if (D)
			Log.d(TAG, "connected");

		// Cancel the thread that completed the connection
		if (mConnectThread != null) {
			mConnectThread.cancel();
			mConnectThread = null;
		}

		// Cancel any thread currently running a connection
		if (mConnectedThread != null) {
			mConnectedThread.cancel();
			mConnectedThread = null;
		}
```

```
            // Cancel the accept thread because we only want to
connect to one
            // device


            // Start the thread to manage the connection and perform
transmissions

            mConnectedThread = new ConnectedThread(socket);

            mConnectedThread.start();

            // Send the name of the connected device back to the UI
Activity
            Message msg =
mHandler.obtainMessage(MainActivity.MESSAGE_DEVICE_NAME);
            Bundle bundle = new Bundle();
            bundle.putString(MainActivity.DEVICE_NAME,
device.getName());
            msg.setData(bundle);
            mHandler.sendMessage(msg);

            setState(STATE_CONNECTED);
```

6. ZAKLJUČAK

U ovom radu opisano je izrađivanje danas sve više popularne Android aplikacije. Na samom početku prije odabira zadatka, sve češćim korištenjem aplikacija izazvalo je zanimanje za samostalnom izradom Android aplikacije. Sama izrada grafičkog sučelja nije teška za izraditi ali pisanje programa na temelju kojeg i radi aplikacija je vrlo zahtjevno. Potrebno je dobro poznavati programski jezik Java koji sam morao učiti tijekom svoje izrade Android aplikacije. Kao i svaki drugi programski jezik pa tako i Java koristi mnogo naredbi gdje svaka od naredbi izvršava određeni zadatak. Aplikacija je izrađena samo za upravljanje vanjskim koordinatama robota. Aplikaciju je moguće proširiti tako da se omogući i upravljanje unutarnjim koordinatama robota. Na taj način bi povećali bolju upravljivost robota te bolje pozicioniranje kad upravljamo robotom preko mobilnog uređaja. Program Eclipse u kojem je rađena aplikacija ima svojih prednosti i nedostataka. Za početnike program pruža puno stvari kako bi naučili izrađivati aplikacije. Za izradu aplikacija postoji još mnogo programa te je moje mišljenje da je ovaj program najjednostavniji za početnike. Nadamo se da će se ovom aplikacijom malo modernizirati robot, te da će studenti u budućnosti upravljati robotom preko mobilnih uređaja.

LITERATURA

[1] https://bib.irb.hr/datoteka/578851.Final_Crnekovic_Zorc-Kinematic_controller.pdf

[2] http://e-ucenje.fsb.hr/course/view.php?id=870 Robot Mitsubishi-RM501

[3] http://www.roboex.com/pdf/Mitsubishi%20Robot%20Manual%20RM-501.pdf

[4] http://en.wikipedia.org/wiki/Bluetooth

[5] http://developer.android.com/guide/topics/connectivity/bluetooth.html

[6] https://lh6.googleusercontent.com/mxwXq2r6V0TIBeS_sWwyW93ifKbORiugbxe

[7] http://www.robotshop.com/media/catalog/product/cache/1/image/800x800/9df78eab33525d08d6e5fb8d27136e95/3/5/350-bluetooth-mate-gold-module.jpg

[8] Tugomir Šurina, Mladen Crneković. Industrijski roboti. Sveučilište u Zagrebu, Školska knjiga, 1990.

[9] http://sr.scribd.com/doc/36338820/BT-skripta

[10] http://hr.wikipedia.org/wiki/Android_(operacijski_sustav)

[11] http://4.bp.blogspot.com/_GdmyogjL4Jc/TDlkwJmIw6I/AAAAAAAAAC0/tBvNX5dt2P8/s1600/Bluetooth%2BLogo.PNG

[12] http://www.fer.unizg.hr/_download/repository/Android(2009)Jukic_Ivan.pdf

[13] http://android.vidilab.com/2010/04/sto-je-android/

[14] http://fin6.com/2013/07/android-2014/

[15] https://source.android.com/

[16] http://cdn.droidviews.com/wp-content/uploads/2014/01/Android-4.4.2-KitKat-Galaxy-S4.jpg

[17] http://www.govtedu.com/wp-content/uploads/ebooks/android/the-android-developers-cookbook.pdf

[18] http://it-ebooks.info/book/621/

PRILOZI

I. CD-R disc